Set Covering-based Approximation Algorithm for Delay Constrained Relay Node Placement in Wireless Sensor Networks

Chaofan Ma,Wei Liang, *Member, IEEE,* and Meng Zheng, *Member, IEEE,*

***Abstract*—The Delay Constrained Relay Node Placement (DCRNP) problem in Wireless Sensor Networks (WSNs) aims to deploy minimum relay nodes such that for each sensor node there is a path connecting this sensor node to the sink without violating delay constraint. As WSNs are gradually employed in time-critical applications, the importance of the DCRNP problem becomes noticeable. For the NP-hard nature of DCRNP problem, an approximation algorithm-Set-Covering-based Relay Node Placement (SCA) is proposed to solve the DCRNP problem in this paper. The proposed SCA algorithm deploys relay nodes iteratively from sink to the given sensor nodes in hops, i.e., in the kth iteration SCA deploys relay nodes at the locations that are k hops apart from the sink. Specifically, in each iteration, SCA first finds the candidate deployment locations located within 1 hop to the relay nodes and sensor nodes, which have already been connected to the sink. Then, a subset of these candidate deployment locations, which can guarantee the existence of paths connecting unconnected sensor nodes to the sink within delay constraint, is selected to deploy relay nodes based on the set covering method. As the iteration of SCA algorithm, the sensor nodes are gradually connected to the sink with satisfying delay constraint.**

The elaborated analysis of the approximation ratio of SCA algorithm is given out, and we also prove that the SCA is a polynomial time algorithm through rigorous time complexity analysis. To evaluate the performance of the proposed SCA algorithm, extensive simulations are implemented, and the simulation results show that the SCA algorithm can significantly save the deployed relay nodes comparing to the existing algorithms, i.e., at most 31.48% deployed relay nodes can be saved due to SCA algorithm.

***Index Terms*—Wireless sensor networks, set covering, delay constrained relay node placement, approximation algorithm.**

I. Introduction

Wireless Sensor Networks (WSNs) attract considerable attention in recent years for their immense potential in the applications of environment monitoring, battlefield reconnaissance and industrial automation [1]-[2]. WSNs comprise spatially distributed Sensor Nodes (SNs), which sense certain circumstance information, and one or several sinks to collect the information sensed by SNs. Typically, SNs are cheap and powered by batteries. This results in that the power and communication radii of a SN is limited. Thus, to prolong the network lifetime and improve the scalability, Relay Nodes (RN) are introduced into WSNs. RNs are equipped with plenty power and large communication radii, however, the RNs are also costly [3]-[4]. Hence, the Relay Node Placement (RNP) problem try to build connected WSNs by deploying a minimum number of RNs subject to various constraints, such as network lifetime, throughput and delay.

Recently, the importance of Delay Constrained RNP (DCRNP) problem is highlighted by the employment of WSNs in the time-critical applications, e.g., the factory automation and patent monitoring [5]-[6]. In the factory environment, the data sensed by SNs is typically time-sensitive, such as alarm notification and information for feedback control, and thus the importance of receiving the data at the sink in a timely manner is noticeable [7]-[8]. It has been reported in [19] that DCRNP problem is NP-hard, which implies that no polynomial time algorithm exists for this problem as long as $P \neq NP$.

Although RNP problem has been extensively studied, the literature about DCRNP problem is rare. Bhattacharya and Kumar [19]-[21] first proved that DCRNP problem is NP-hard, and then proposed a polynomial time approximation algorithm to solve this problem. Nigam and Agarwal [23] designed an algorithm to optimally solve DCRNP problem without guaranteeing a polynomial time complexity. As in [19]-[23], in this paper, RNs can only be placed at a subset of the predetermined locations, which are called the Candidate Deployment Locations (CDLs).

In order to devise a polynomial time algorithm with better performance, a Set-Covering-based Approximation (SCA) algorithm is proposed in this paper to yield a desirable solution for DCRNP problem. The proposed SCA is an iterative algorithm and deploys RNs in hops from the sink to SNs. In order to maintain network connectivity, in each iteration, the RNs that are deployed based on set covering method have 1-hop neighbor SNs or RNs, which have been connected to the sink within delay constraint. As SCA algorithm is iteratively executed, SNs are gradually connected to the sink with fulfilling delay constraint. To evaluate the proposed SCA algorithm, extensive simulations are implemented, and the simulation results show that the SCA algorithm outperforms the existing algorithms outstandingly.

The major contributions of this paper are listed as follows:

- An approach is designed to represent each CDL or SN by the unconnected SNs, which can be connected to the sink

This work was supported by the Natural Science Foundation of China (61174026, 61233007, and 61304263) and Cross-disciplinary Collaborative Teams Program for Science, Technology and Innovation of Chinese Academy of Sciences (Network and System Technologies for Safety Monitoring and Information Interacting in Smart Grid).

Chaofan Ma, Wei Liang, and Meng Zheng are with the Key Laboratory of Networked Control Systems, Chinese Academy of Sciences, Shenyang 110016, China. E-mail: {machaofan, weiliang, zhengmeng_6}@sia.cn.

Chaofan Ma is also with the University of Chinese Academy of Sciences, Beijing 100049, China.

by the paths that fulfil delay constraint and pass through this CDL or SN.

- An approximation algorithm-SCA is proposed for DCRNP problem. In each iteration, SCA selects a subset of the CDLs, which are located within 1-hop from the RNs and SNs that have been connected to the sink, to deploy RNs by solving a set covering problem that are modeled by the approach mentioned above. Due to this manner, SCA can avoid the limitation suffering by the algorithm presented in [19]-[21].
- The approximation ratio of SCA algorithm is explicitly analyzed, and the time complexity of SCA algorithm is also detailed in this paper.

This paper is organized as follows. Section II reviews the related works. The problem formulation is given out in Section III. The preliminaries and heuristic are given out in Section IV. Section V describes our proposed algorithm for the DCRNP problem. Section VI shows an explicit analysis of approximation ratio and time complexity of the proposed heuristic algorithm. The efficiency of the proposed algorithm is validated through extensive simulations in Section VII. Finally, Section VIII concludes the whole paper.

II. Related Work

The related literatures are divided into two categories, i.e., the literatures about the DCRNP problem and the RNP problem without delay constraint.

A. RNP Problem without Delay Constraint

Lin and Xue [9] studied the RNP problem and formulated it as the as the Steiner minimum tree with minimum number of Steiner points and bounded edge length problem (SMT-MSP). Then, they proved the SMT-MSP to be NP-Complete and proposed a 5-approximation algorithm. Chen et al. [10] demonstrated that the algorithm proposed in [9] is actual a 4-approximation algorithm, and proposed a 3-approximation algorithm for the RNP problem. Cheng et al. [11] presented a 3-approximation algorithm and a 2.5-approximation algorithm based on so-called 3-star structure. Lloyd and Xue [13] presented a 7-approximation algorithm for the single-tired network RNP problem and a $(5+\epsilon)$-approximation algorithm for the two-tired network RNP problem. Srinivas et al. [14] studied the problem of constructing and maintaining the wireless backbone network for the WSNs, and they also considered mobile wireless backbone network.

Misra et al. [16]-[17] studied the constrained RNP problem in the single-tiered network and proposed a polynomial time O(1)-approximation algorithm. Yang et al. [18] first studied the constrained RNP problem in the two-tiered network and proposed an algorithm with O(1)-approximation ratio for 1-connected single cover problem. Then they proposed two algorithms with O(1)-approximation ratio and O(ln$n$)-approximation ratio under different settings for 2-connected double cover problem.

B. DCRNP Problem

The DCRNP problem has been studied in [19]-[23]. Bhattacharya and Kumar [19]-[21] first proved that the DCRNP problem is NP-hard, and a shorted path tree based algorithm is proposed. This algorithm preliminarily forms a shortest path tree to connect each SN to the sink, then it saves the deployed RNs by gradually removing the RNs on the shortest path tree. This brings in a limitation that the deployed RNs can only be those contained in the originally yielded shortest path tree, and the worst case of this algorithm happens when all the RNs of the optimal solution have been missed by the shortest path tree. Sitanayah et al. [22] studied the fault-tolerant RNP problem under the length constraint, and proposed a heuristic to solve this problem. However, no time complexity analysis and performance guarantee were given. Nigam and Agarwal [23] formulated the DCRNP problem as a linear programming problem, and proposed a branch-and-cut algorithm to optimally solve the DCRNP problem. However, the proposed algorithm can only solve a special case of the DCRNP problem (each of the source node cannot have a singleton node cut), and the time complexity of this algorithm grows exponentially, which indicates it cannot be applied to the large scale problem.

Some researches on the related scopes are also listed in follows. Voss [24] used the hop count to represent the network delay, and studied the hop constrained Steiner tree problem. A local search algorithm based on tabu search was proposed, but this paper did not analyze the approximation ratio and the time complexity. Costa et al. [25] studied the Steiner tree problem with revenue, budget and hop constraints. A tabu search based heuristic was proposed to solve this problem, but the proposed heuristic is not a polynomial time algorithm. Gouveia et al. [26] studied the distance constrained minimum spanning tree problem, and three approaches were proposed based on the column generation scheme and the Lagrangian relaxation. However, no time complexity analysis is provided in the paper.

III. Problem Formulation

The network delay is typically composed of the processing delay, the queuing delay, the transmission delay and the propagation delay. In the WSNs, the distance between two different SNs is normally a few tens or hundreds of meters. Therefore, normally compared to the other three kinds of delay, the transmission delay can be ignored. Since the rest of delays are at proportional to the hop count, in this paper, the network delay is represented by the hop count. Therefore, DCRNP problem is reduced to a hop constrained RNP problem in the rest of this paper.

Given a set of SNs $S=\{s_1, s_2, ..., s_n\}$, a set of Candidate Deployment Locations (CDLs) $C=\{c_1, c_2, ..., c_m\}$ for deploying RNs and a sink K, we can build an undirected graph $G=\{V, E\}$, where $V=S\bigcup C\bigcup\{K\}$ is the node set, and E is the edge set. $\forall u, v \in V$ $(u \neq v)$, if u and v are the two ends of an edge in E, u and v should fulfil the following conditions:

- If $u \in S$ or $v \in S$, then u and v should meet that $\|u - v\| \leq r$;
- if $u \notin S$ and $v \notin S$, then u and v should meet that $\|u - v\| \leq R$,

where $\|u - v\|$ denotes the Euclidean distance between u and v. r and R ($r \leq R$) are the communication radii of the SN and RN, respectively. Besides, in this paper a path between u and v is denoted by $p(u, v)$.

Definition 1 (DCRNP Problem): For an undirected graph $G = \{V, E\}$, the DCRNP problem searches for an induced subgraph $G' = \{V', E'\}$ of G, where $V' = S \bigcup C' \bigcup \{K\}$, and C' is a subset of C with the minimum cardinality such that the following condition is satisfied: there exists at least one path, which complies with the delay constraint, between each SN and the sink.

Let $P_i = \{p_1(s_i, K), p_2(s_i, K), ..., p_{k_i}(s_i, K)\}$ $(1 \leq i \leq n)$ be the k_i paths connecting the SN s_i and the sink K, where $p_j(s_i, K)$ $(1 \leq j \leq k_i)$ denotes the jth path between s_i and K. Let $P = \bigcup\limits_{i=1}^{n} P_i$ denote the set of all the paths connecting the given SNs and the sink K.

CDLs on the induced subgraph G' are selected to deploy RNs, thus, the terms RN and CDL are interchangeable in this paper. Consequently, a node on G' can be a SN, RN or the sink. Let $\mathcal{N}(p)$ denote all the nodes on path p. Then, let $\mathbb{C}(p)$ denote RNs on path p, and the notation $\mathcal{C}(p)$ represent the hop count of path p. Obviously, we have that $\mathcal{C}(p) = |\mathcal{N}(p)| - 1$.

The DCRNP problem can be formulated as an optimization problem:

$$\text{Minimize} \quad |C'| \tag{1a}$$

$$\begin{aligned} s.t. \quad & \forall P_i \in P \ (1 \leq i \leq n), \\ & \exists p_j(s_i, K) \in P_i \ (1 \leq j \leq |P_i|), \mathcal{C}\left(p_j(s_i, K)\right) \leq \Delta_i \end{aligned} \tag{1b}$$

where Δ_i is the delay constraint for the ith SN. Without loss of generality, in this paper we assume that the given SNs have the same delay constraint, i.e., $\Delta_i = \Delta$, $i \in \{1, 2, ..., n\}$.

The DCRNP problem has been proved NP-hard [19], therefore, this paper proposes a heuristic algorithm to approximately solve the DCRNP problem. The path connecting the ith SN s_i and the sink and satisfying the delay constraint Δ is called a **feasible path** of s_i, and the subgraph satisfying the formulation (1b) is called a **feasible graph** of the DCRNP problem.

IV. Preliminaries and Heuristic

Feasible graphs for a DCRNP problem should be connected. Thus, we can find an induced subgraph, which is a tree rooted at the sink and connecting all the SNs, from each feasible graph. Therefore, in this paper, the proposed algorithm yields a feasible graph that is a tree for each given DCRNP problem.

Given a tree T and two different nodes of T, u and v, we denote $p_T(u, v)$ as a path from u to v of T. $\forall s \in S$, if $\mathcal{C}(p_T(K, s)) \leq \Delta$, we call T a feasible tree for the given DCRNP problem.

Next, the conception of the level of a tree will be introduced to facilitate the explanation of the proposed SCA algorithm. Given a feasible tree T, the level of a node q in T is the hop count of the path between the sink K and node q, i.e., $\mathcal{C}(p_T(K, q))$. The set of nodes on the kth level of T is denoted by L_T^k. As illustrated in Fig. 1, $L_T^1 = \{c_1, c_5\}$, $L_T^2 = \{c_2, c_3, c_6\}$, $L_T^3 = \{s_1, s_4, c_4, c_7\}$ and $L_T^4 = \{s_2, s_3\}$.

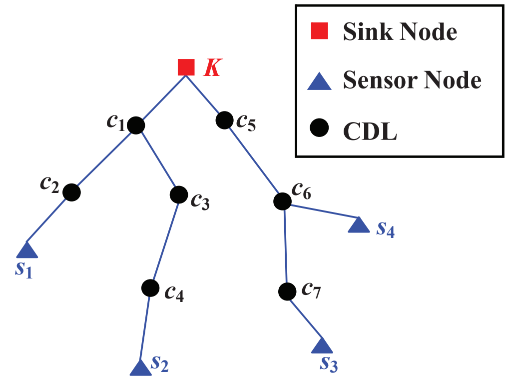


Fig. 1. An illustration of the conception of the level.

Observing that the nodes on each level should have 1-hop neighbors on their adjacent levels to maintain the network connectivity, we can deploy RNs from sink to SNs level by level such that the sink is gradually connected to SNs by the deployed RNs. Suppose that we have deployed RNs from the 1st level to the $(k-1)$th level of T, and now attempt to deploy RNs for the kth level. Let $\mathcal{V}(L_T^{k-1})$ denote the set of 1-hop neighbors of the nodes in L_T^{k-1}. Obviously, the nodes on kth level should be 1-hop neighbors of nodes on $(k-1)$th level, i.e., $L_T^k \subseteq \mathcal{V}(L_T^{k-1})$.

Therefore, we move on to search the 1-hop neighbors of the nodes on $(k-1)$th level. As the search is accomplished, i.e., $\mathcal{V}(L_T^{k-1})$ is known, we try to select the nodes on kth level from $\mathcal{V}(L_T^{k-1})$. The nodes that are selected to be located on the kth level should guarantee the existence of feasible paths for the unconnected SNs. For this purpose, an approach, which represents each node q in $\mathcal{V}(L_T^{k-1})$ by the unconnected SNs that can be connected to the sink by the feasible paths passing through q, is designed and shown in the following path of this section.

In this paper, the shortest path between u and v means a path connecting u and v with least hop count, and the notation $\mathcal{H}(u, v)$ stands for a shortest path between u and v. Let $\bar{P}_i$ $(1 \leq i \leq n)$ be the set of all feasible paths between a SN s_i and the K, i.e., $\forall \mathcal{C}(p(s_i, K)) \leq \Delta$, $p(s_i, K) \in \bar{P}_i$. Consequently, the set of nodes lying on the feasible paths of s_i is $\bigcup\limits_{x \in \bar{P}_i} \mathcal{N}(x)$. Let q be an arbitrary CDL or SN.

Theorem 1: The sufficient and necessary condition for $q \in \bigcup\limits_{x \in \bar{P}_i} \mathcal{N}(x)$ is that

$$\mathcal{C}(\mathcal{H}(q, s_i)) + \mathcal{C}(\mathcal{H}(q, K)) \leq \Delta. \tag{2}$$

Proof: Sufficiency: From $q \in \bigcup\limits_{x \in \bar{P}_i} \mathcal{N}(x)$ we knows that there is at least one path, p ($p \in \bar{P}_i$), passing through q, therefore, we have $\mathcal{C}(p) \leq \Delta$.

Let $\bar{p}$ ($\tilde{p}$) be a segment of p between q and s_i (K). Then, we have

$$\mathcal{C}(\bar{p}) \geq \mathcal{C}(\mathcal{H}(q, s_i)) \tag{3}$$

and

$$\mathcal{C}(\tilde{p}) \geq \mathcal{C}(\mathcal{H}(q, K)). \tag{4}$$

Since p consists of $\bar{p}$ and $\tilde{p}$, p is a feasible path, which implies

$$\mathcal{C}(p) = \mathcal{C}(\bar{p}) + \mathcal{C}(\tilde{p}) \leq \Delta. \tag{5}$$

Plunging inequalities (3) and (4) into (5), we can achieve

$$\mathcal{C}(\mathcal{H}(q, s_i)) + \mathcal{C}(\mathcal{H}(q, K)) \leq \Delta. \tag{6}$$

So far, we have completed the first part of the proof.

Necessity: We build two shortest paths which connect q and s_i and connect q and K, respectively, i.e., $\mathcal{H}(q, s_i)$ and $\mathcal{H}(q, K)$. Then, we can form a path p between s_i and K by combining $\mathcal{H}(q, s_i)$ and $\mathcal{H}(q, K)$. According to the inequality (2), the hop count of p is given by

$$\begin{aligned}\mathcal{C}(p) &= \mathcal{C}(\mathcal{H}(q, s_i)) + \mathcal{C}(\mathcal{H}(q, K)) \\ &\leq \Delta,\end{aligned} \tag{7}$$

which confirms that p is a feasible path between s_i and K. Therefore, we can conclude that

$$q \in \bigcup_{x \in \bar{P}_i} \mathcal{N}(x). \tag{8}$$

This completes the proof of Theorem 1. ■

Theorem 1 implies that if a RN or SN q lies on a feasible path of a SN, then q should satisfy inequality (2). Let $\bar{S}_k$ be the set of unconnected SNs, which are located on the levels higher than k. As mentioned above, we expect to select the nodes on the kth level of T from $\mathcal{V}(L_T^{k-1})$ such that feasible paths exist for the unconnected SNs. To this end, for each node q in $\mathcal{V}(L_T^{k-1})$, we try to find the unconnected SNs, which can be connected to sink by the feasible paths passing through q, and the set of these SNs is denoted by $\mathcal{Q}(q)$. According to Theorem 1, the SNs in $\mathcal{Q}(q)$ should fulfil that

$$\mathcal{C}(\mathcal{H}(s, q)) + k \leq \Delta, \tag{9}$$

where $s \in \mathcal{Q}(q)$, and we say that s is connected to the sink by q.

When q ($q \in \mathcal{V}(L_T^{k-1})$) is selected as a node on the kth level, according to Theorem 1, we can promise the existence of feasible paths, which pass through q and connect the SNs in $\mathcal{Q}(q)$ to the sink within delay constraint.

However, the inequality (9) cannot ensure that the RNs deployed on current level are closer to the unconnected SNs than the nodes on previous levels, which may bring in a large amount of redundant deployed RNs. As shown in Fig. 2, delay constraint is set as $\Delta = 7$. According to inequality (9), we can obtain that $\mathcal{Q}(c_1) = \{s_1, s_2, s_3\}$ and $\mathcal{Q}(c_2) = \{s_2, s_3\}$. Since $\mathcal{V}(L_T^0) = \{c_1, c_2\}$ and all the SNs can be connected to the sink by c_1, to select as fewer CDLs as possible on each level, only c_1 is selected to be allocated on the 1st level. This ignores that the feasible path that passes through c_1 and connects s_2 and s_3 to the sink is much longer than the feasible path passing through c_2 to s_2 and s_3, i.e., $\mathcal{C}(p(c_1, s_2)) > \mathcal{C}(p(c_2, s_2))$ and $\mathcal{C}(p(c_1, s_3)) > \mathcal{C}(p(c_2, s_3))$. This ignorance leads to a local optimum, whose edges are denoted by the blue dashed line segments in Fig. 2, and misses the optimal solution, whose

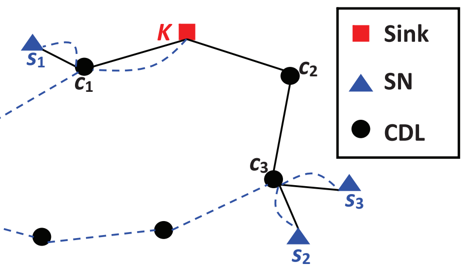


Fig. 2. An illustration of the redundant RNs brought by the utilization of inequality (9).

edges are represented by the black line segments. As a result, totally 3 redundant RNs are introduced.

To this end, we introduce a rule to guarantee that the RNs deployed on each level gradually move closer to the unconnected SNs. Let $\bar{q}$ be a node on the $(k-1)$th level, and q be a 1-hop neighbor of $\bar{q}$.

Lemma 1: $\mathcal{Q}(q) \subseteq \mathcal{Q}(\bar{q})$.

Proof: For each SN in $\mathcal{Q}(q)$, the inequality (9) is met, i.e.,

$$\forall s \in \mathcal{Q}(q),\ \mathcal{C}(\mathcal{H}(s, q)) + k \leq \Delta. \tag{10}$$

Since q is a 1-hop neighbor of $\bar{q}$ and $\bar{q}$ is a node on the $(k-1)$th level, the following inequality is fulfilled

$$\forall s \in \mathcal{Q}(q),\ \exists p(s, \bar{q}),\ \mathcal{C}(p(s, \bar{q})) + k - 1 \leq \Delta. \tag{11}$$

Due to the fact that $\mathcal{C}(p(s, \bar{q})) \geq \mathcal{C}(\mathcal{H}(s, \bar{q}))$, we can conclude that

$$\forall s \in \mathcal{Q}(q),\ \mathcal{C}(\mathcal{H}(s, \bar{q})) + k - 1 \leq \Delta. \tag{12}$$

which implies that $\forall s \in \mathcal{Q}(q)$, $s \in \mathcal{Q}(\bar{q})$. Thus $\mathcal{Q}(q)$ is a subset of $\mathcal{Q}(\bar{q})$. This completes the proof. ■

According to Lemma 1, the SNs in $\mathcal{Q}(q)$ can also be connected to the sink by $\bar{q}$. Therefore, to guarantee that the nodes on the kth level are closer to the unconnected SNs than their 1-hop neighbors on the $(k-1)$th level, we restrict the nodes on the kth level to satisfy a rule

$$\mathcal{C}(\mathcal{H}(q, s)) < \mathcal{C}(\mathcal{H}(\bar{q}, s)), \tag{13}$$

where $s \in \mathcal{Q}(q)$ and $s \in \mathcal{Q}(\bar{q})$. Finally, the SNs in $\mathcal{Q}(q)$ should meet both the inequalities (9) and (13).

As illustrated in Fig. 3, c_1 is the only node on the 1st level, and s_1 is connected to the sink by c_1. Additionally, the delay constraint is set as 6, and $\mathcal{C}(\mathcal{H}(c_1, s_1)) = 4$. The 1-hop neighbors of c_1 are represented by the green circles, i.e., c_2, c_3, c_4, c_5 and c_6. The shortest paths from these 1-hop neighbors to s_1 are represented by the green dashed line segments. It can be seen from Fig. 3 that the inequality (9) is met by these 1-hop neighbors except for c_6, but only c_3 and c_4 have a hop count to s_1 less than 4. Thus, in these 1-hop neighbors, only c_3 and c_4 fulfil both the formulations (9) and (13), i.e., s_1 can be connected to the sink only by c_3 and c_4.

To connect all the unconnected SNs to the sink, we should select a subset U of $\mathcal{V}(L_T^{k-1})$ such that each unconnected SN can be connected to the sink by these nodes, i.e., $\bigcup_{u \in U} \mathcal{Q}(u) =$

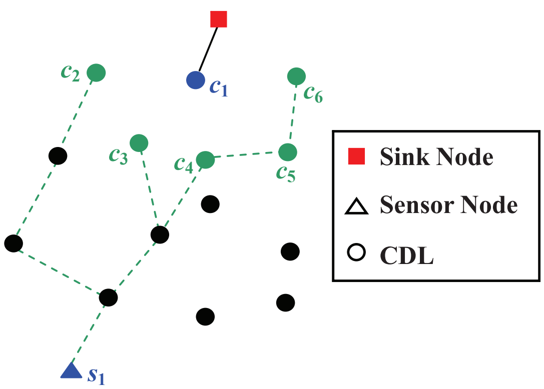


Fig. 3. An illustration of how to determine the SNs connected to the sink by a CDL.

$\bar{S}_k$. Furthermore, to save the deployed RNs we expect to minimize the cardinality of subset U. Thus, we can formulate the problem, which requires to select the nodes on kth level from $\mathcal{V}(L_T^{k-1})$, as the set covering problem described follows: given an unconnected SNs set $\bar{S}_k$ and a node set $\mathcal{V}(L_T^{k-1})$, whose each element q is represented by the unconnected SNs that can be connected to the sink by q, then this problem searches for a minimum subset U of $\mathcal{V}(L_T^{k-1})$ such that $\bigcup\limits_{u \in U} \mathcal{Q}(u) = \bar{S}_k$. As a result, we can employ the set covering algorithm to solve this problem. Due to the effective performance and the wide utilization, the greedy heuristic algorithm is employed to solve this set covering problem.

V. Description of SCA

The proposed SCA algorithm is composed of three steps and detailed in Algorithm 1. In the first step, the SCA algorithm preliminarily tests either a feasible solution exists or RNs are necessary for the DCRNP problem based on shortest path tree algorithm.

The second step is the principal body of the SCA algorithm. The second step of SCA is iteratively executed so as to allocate the nodes on each level. To be specific, in the kth iteration, SSCA moves on to select nodes on the kth level. Firstly, the second step of SCA searches for 1-hop neighbors of the nodes on the $(k-1)$th level, i.e., $\mathcal{V}(L_T^{k-1})$. Next, for each node q in $\mathcal{V}(L_T^{k-1})$, the second step of SCA searches for the unconnected SNs, which can be connected to the sink by q, i.e., $\mathcal{Q}(q)$. Then, each node q in $\mathcal{V}(L_T^{k-1})$ is represented by $\mathcal{Q}(q)$, and as a result the current problem is transformed into the set covering problem, which attempts to select a minimum subset of $\mathcal{V}(L_T^{k-1})$ to fully cover the unconnected SNs. Finally, the greedy heuristic algorithm is employed to solve this set covering problem. Consequently, the selected subset of $\mathcal{V}(L_T^{k-1})$ is the set of nodes on kth level. As the execution of the second step of SCA, a feasible tree T is gradually built, and clearly, due to the delay constraint, the number of iteration of the second step of SCA cannot be larger than Δ.

The process of the second step of SCA is illustrated in Fig. 4, where the circles and triangles with the same color are the 1-hop neighbors of the nodes on the same level, and the circles and triangles with red edges are the SNs or RNs on each level. At the beginning, the second step of SCA first searches the 1-hop neighbors of the sink K, and these neighbors are denoted by the black circles and triangles. Next, the second step of SCA represents these neighbors by the unconnected SNs, and the greedy heuristic is employed to find a set cover for the unconnected SNs. As shown in Fig. 4, c_1 is selected in this iteration. In the second iteration, the second step of SCA first finds the 1-hop neighbors of c_1, and these neighbors are denoted by the blue circles and triangles. Then, s_1 and c_3 are selected in this iteration. As the iteration repeats, all the unconnected SNs are gradually connected to the sink. The second step of SCA is detailed in the Algorithm 2.

Algorithm 1 SCA.

Require:
A set of SNs $S = \{s_1, s_2, ..., s_n\}$, a set of CDLs $C = \{c_1, c_2, ..., c_m\}$, a sink node K, the communication radii of the SN r and the RN R, the delay constraint Δ.

Ensure:
A feasible tree T.

tmp =a shortest path tree rooted at K and including all SNs in S with using CDLs in C; %the first step of SCA begins.
if $\forall i \in \{1, 2, ..., n\}$, $p_{tmp}(s_i, K) \le \Delta$ **then**
  tmp =a shortest path tree rooted at K and including all SNs in S without any CDLs in C;
  if $\exists i \in \{1, 2, ..., n\}$, $p_{tmp}(s_i, K) > \Delta$ **then**
    input S, C and K into the second step of SCA; %the first step of SCA ends.
    C' =a subset of C returned by the second step of SCA;
    input S, C' and K into the third step of SCA;
    T =a feasible tree returned by the third step of SCA;
  else
    declare that no RNs are necessary to build a feasible tree T and terminate SCA;
  end if
else
  declare that no feasible solutions exist for this DCRNP problem and terminate SCA;
end if
return T;

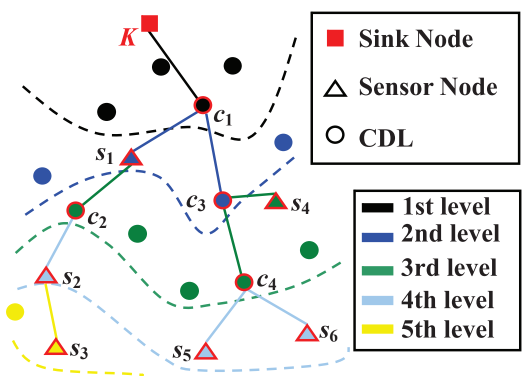


Fig. 4. An illustration of the procedure of the second step of SCA.

Remark 1: As 1-hop neighbors have been represented by the SNs connected to the sink by them, the situation that several neighbors connect the same unconnected SNs to the sink may happen, i.e., $\mathcal{Q}(q) = \mathcal{Q}(\bar{q})$, where q and $\bar{q}$ are all 1-hop neighbors of the nodes on the last level. To deal with this situation, the weight of a neighbor, q, is introduced and defined as follows. Let $\mathcal{Q}(q)$ be the set of unconnected SNs,

which are connected to the sink by q. Then the weight, $\omega(q)$, of q is given by

$$\omega(q) = \left| \bigcup_{x \in \mathcal{Q}(q)} \mathbb{C}(\mathcal{H}(q,x)) \right|. \tag{14}$$

If the situation mentioned above happens, we select the one with least weight.

Algorithm 2 The Second step of SCA.

Require:
A set of SNs $S = \{s_1, s_2, ..., s_n\}$, a set of CDLs $C = \{c_1, c_2, ..., c_m\}$, a sink node K, the communication radii of the SN r and the RN R, the delay constraint Δ.
Ensure:
A set, C', of CDLs.

$\mathcal{V}(L_T^0)$ = the 1-hop neighbors of K in S and C;
$L_T^0 = K$;
$k = 1$;
$\bar{S}_k = S$;
while ($\bar{S} \neq \varnothing$) **do**
  $\bar{S}_k = \bar{S}_k - \mathcal{V}(L_T^0)$;
  $i = 0$;
  tmp =the ith element in $\mathcal{V}(L_T^0)$;
  while ($tmp \neq \varnothing$) **do**
    $con[i] = \mathcal{Q}(tmp)$;
    $weight[i] = \omega(tmp)$;
    $i = i + 1$;
    tmp =the ith element in $\mathcal{V}(L_T^0)$;
  end while
  sort $\mathcal{V}(L_T^0)$ according to the cardinality of each $con[i]$ and the $weight[i]$;
  L_T^k = a subset of $\mathcal{V}(L_T^0)$ found by the greedy heuristic algorithm;
  $\bar{S}_k = \bar{S}_k - L_T^k$;
  $C = C - L_T^k$;
  $\mathcal{V}(L_T^k)$ = the 1-hop neighbors, which are selected from $\bar{S}_k$ and C, of the nodes in L_T^k;
  $tmpRe = L_T^k - S$;
  $C' = C' \bigcup tmpRe$;
  $k = k + 1$;
end while
return C';

The redundant RNs may be deployed by the second step SCA for the introduction of the rule (13), which is illustrated in Fig. 5. The feasible tree returned by the second step SCA is denoted by the black line segments in Fig. 5. Since $\mathcal{C}(\mathcal{H}(K, s_1)) < \mathcal{C}(\mathcal{H}(c_3, s_1))$, s_1 cannot be connected to the sink by c_3, and the second step SCA select two CDLs to deploy RNs, i.e., c_1 and c_3, on the first level. However, there is a feasible path passing through c_3 and connecting K to s_1. Thus, the redundant RNs c_1 and c_2 are introduced. Actually, a solution with fewer RNs exists in Fig. 5, and this solution is represented by the red dashed line segments.

To save the redundant RNs introduced by the second step of SCA, the third step of the SCA algorithm is designed. Since we expect that the number of RNs deployed on each level is as fewer as possible, we try remove the RNs connecting fewer SNs to the sink on each level. Thus, each RN q returned by the second step of SCA is assigned a weight, which is defined as the number of SNs meeting the inequality (2) for q. Then, the returned RNs are sorted in ascending order according to the weight, and we try to gradually delete the redundant RNs from the one with least weight. As a RN is deleted, the shortest path tree is formed to check whether a feasible tree exists for the DCRNP problem on the remaining RNs. If such feasible tree exists, the RN is deleted forever, otherwise the RN is retrieved and marked as tried. This procedure is repeated until all the RNs is deleted or marked as tried. The third step of the SCA algorithm is detailed in the Algorithm 3.

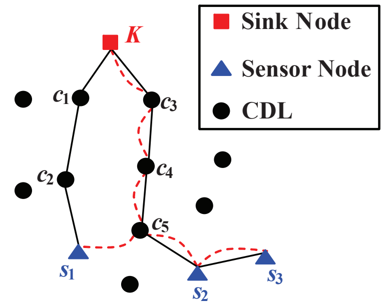


Fig. 5. An illustration of the redundant CDLs introduced by the second step SCA.

Algorithm 3 The Third step of SCA.

Require:
The set S of SNs, a set C' of CDLs returned by the second step of SCA.
Ensure:
A feasible tree T.

T = a shortest path tree rooted at the sink and including all the given SNs by using the CDLs in C';
for all $tmp \in C'$ **do**
  calculate the weight of tmp;
end for
sort the CDLs in C' in an ascending order according the weight;
tmp = the first element of C';
$C' = C' - tmp$;
while ($tmp \neq \varnothing$) **do**
  $tmpT$ = a shortest path tree rooted at the sink and including all the given SNs by using the CDLs in C';
  if ($tmpT$ is a feasible tree) **then**
    T = $tmpT$;
    C' = the CDLs on T;
    sort the CDLs in C' in an ascending order according the weight;
    tmp = the first element of C';
    $C' = C' - tmp$;
  else
    mark tmp as tried;
    $C' = C' \bigcup tmp$;
    tmp = the least weighted untried CDL in C';
  end if
end while
return T;

VI. Analysis of SCA

A. Approximation Ratio

Let T^* be a optimal feasible tree for the given DCRNP problem, and the set of the RNs and SNs on the kth level of T^* is denoted by $L_{T^*}^k$. Let T be a feasible tree returned by the SRNP algorithm, and the set of RNs and SNs on the kth level of T is denoted by $L_{T^*}^k$. Thus, the ratio between the optimal solution and the solution returned by the SCA algorithm is given by

$$\frac{\left| \bigcup_{k=1}^{l} L_T^k - S \right|}{\left| \bigcup_{k=1}^{l^*} L_{T^*}^k - S \right|} = \frac{\left| \bigcup_{k=1}^{l} (L_T^k - S) \right|}{\left| \bigcup_{k=1}^{l^*} (L_{T^*}^k - S) \right|}, \tag{15}$$

where l and l^* denote the number of levels of T and T^*, respectively.

Because of the fact that a node can only be located on a certain level, i.e., $L_T^i \bigcap L_T^j = \varnothing$ and $L_{T^*}^i \bigcap L_{T^*}^j = \varnothing$ $(i \neq j)$, equality (15) changes into

$$\frac{\left|\bigcup_{k=1}^{l}\left(L_T^k - S\right)\right|}{\left|\bigcup_{k=1}^{l^*}\left(L_{T^*}^k - S\right)\right|} = \frac{\sum_{k=1}^{l}\left|L_T^k - S\right|}{\sum_{k=1}^{l^*}\left|L_{T^*}^k - S\right|} = \frac{\sum_{k=1}^{l}\left|L_T^k\right| - |S|}{\sum_{k=1}^{l^*}\left|L_{T^*}^k\right| - |S|}, \tag{16}$$

which leads to that

$$\frac{\sum_{k=1}^{l}\left|L_T^k\right| - |S|}{\sum_{k=1}^{l^*}\left|L_{T^*}^k\right| - |S|} < \frac{\sum_{k=1}^{l}\left|L_T^k\right|}{\sum_{k=1}^{l^*}\left|L_{T^*}^k\right|} = \frac{\left|L_T^1\right|}{\sum_{k=1}^{l^*}\left|L_{T^*}^k\right|} + \frac{\left|L_T^2\right|}{\sum_{k=1}^{l^*}\left|L_{T^*}^k\right|} + ... + \frac{\left|L_T^l\right|}{\sum_{k=1}^{l^*}\left|L_{T^*}^k\right|}. \tag{17}$$

Let OPT_k be a minimum set cover for the kth level of T, then, we can obtain that

$$\frac{|L_T^1|}{\sum_{k=1}^{l^*}\left|L_{T^*}^k\right|} + \frac{|L_T^2|}{\sum_{k=1}^{l^*}\left|L_{T^*}^k\right|} + ... + \frac{|L_T^l|}{\sum_{k=1}^{l^*}\left|L_{T^*}^k\right|} = \frac{|L_T^1|}{|OPT_1|}\frac{|OPT_1|}{\sum_{i=1}^{l^*}\left|L_{T^*}^k\right|} + \frac{|L_T^2|}{|OPT_2|}\frac{|OPT_2|}{\sum_{k=1}^{l^*}\left|L_{T^*}^k\right|} + ... + \frac{|L_T^l|}{|OPT_l|}\frac{|OPT_l|}{\sum_{k=1}^{l^*}\left|L_{T^*}^k\right|}. \tag{18}$$

Since greedy algorithm is employed to solve the set covering problem that requires to select a subset of $V(L_T^{k-1})$ to cover the unconnected SNs $\bar{S}_k$ on the kth level, considering that the approximation ratio of greedy set covering algorithm is $\ln|\bar{S}_k|$ [27], we have that

$$\forall k \in \{1, 2, ..., l\},\ \frac{|L_T^k|}{|OPT_k|} \leq \ln|\bar{S}_k|. \tag{19}$$

In the first step of SCA, we check the existence of the feasible solution without RNs for DCRNP problem, thus, for the DCRNP problem input to the second step, its optimal solution contains at least one RN, i.e.,

$$\left|\bigcup_{k=1}^{l^*} L_{T^*}^k - S\right| \geq 1, \tag{20}$$

and according to equations (15)-(16) this leads to that

$$\sum_{k=1}^{l^*}\left|L_{T^*}^k\right| \geq |S| + 1. \tag{21}$$

Besides, due to the fact that $|OPT_k| \leq |\bar{S}_k| \leq |S|$, combining with inequalities (19)-(21), the formulation (18) changes into

$$\begin{aligned} &\frac{|L_T^1|}{|OPT_1|}\frac{|OPT_1|}{\sum_{k=1}^{l^*}\left|L_{T^*}^k\right|} + \frac{|L_T^2|}{|OPT_2|}\frac{|OPT_2|}{\sum_{k=1}^{l^*}\left|L_{T^*}^k\right|} + ... + \frac{|L_T^l|}{|OPT_l|}\frac{|OPT_l|}{\sum_{k=1}^{l^*}\left|L_{T^*}^k\right|} \\ &\leq l\ln|S|\frac{|S|}{|S|+1} \\ &\leq \Delta\ln|S|. \end{aligned} \tag{22}$$

Finally, the approximation ratio of SCA can be represented by

$$\frac{\left|\bigcup_{k=1}^{l} L_T^k - S\right|}{\left|\bigcup_{k=1}^{l^*} L_{T^*}^k - S\right|} = \mathrm{O}(\ln|S|). \tag{23}$$

B. Time Complexity

Let $N = |S| + |C| + 1$, then the time complexity of the shortest path tree algorithm is given by $N\lg N$ [27]. The time complexity of the first step of the SRNP algorithm can be easily calculated, i.e., $\mathrm{O}(N\lg N)$.

Then we analyze the time complexity of the Algorithm 2. In each iteration of the second step SCA, the main loop between lines 5-23 is executed. Therefore, we first analyze the time complexity to execute the main loop for once. In the inner loop between lines 9-14, the shortest path tree algorithm is applied to find the SNs fulfilling the inequalities (9) and (13). Thus, the time complexity of one iteration of this loop is $\mathrm{O}(N\lg N)$. Additionally, this loop can be iterated at most N times, which leads to that the time complexity of this loop is $\mathrm{O}(N^2\lg N)$. Then, the greedy heuristic algorithm with a time complexity of $\mathrm{O}(N^2)$ [27] is employed to find a minimum set cover for each level. Moreover, in each iteration of the second step of SCA, we will find the 1-hop neighbors within $\mathrm{O}(N^2)$ running time. Hence, the time complexity of one iteration of the main loop is

$$\mathrm{O}(N^2\lg N) + \mathrm{O}(N^2) = \mathrm{O}(N^2\lg N). \tag{24}$$

Since the main loop will be iterated at most Δ times, the time complexity of the second step of SCA is $\mathrm{O}(N^2\lg N)$.

In the main loop of the third step of SCA, the shortest path tree algorithm is implemented at most N times, and the time complexity to sort the remaining CDLs is $\mathrm{O}(N^2)$. Thereby, the time complexity of the third step of SCA is $\mathrm{O}(N^3)$. Since the SCA algorithm is the combination of the three steps, the total time complexity, T_{SCA}, of the SCA algorithm is given by

$$\begin{aligned} T_{SCA} &= \mathrm{O}(N\lg N) + \mathrm{O}(N^2\lg N) + \mathrm{O}(N^3) \\ &= \mathrm{O}(N^3), \end{aligned} \tag{25}$$

which indicates that the proposed SCA is a polynomial time algorithm.

VII. Simulation Results

In the simulations, SNs are randomly placed on a square field with side length of 100 units. To ensure a high probability that the DCRNP problem has a feasible solution, the amount

of CDLs is relative large, i.e., is set as 400, and these CDLs are randomly distributed on the deployment field. During the simulation, the amount of SNs is varying from 10 to 100. The simulation is carried out under both the homogeneous network, i.e., $r = R = 10$, $r = R = 15$ and $r = R = 20$, and the heterogeneous network, i.e., $r = 10$ and $R = 15$. SPTiRP algorithm proposed in [19]-[21] is taken as the baseline. Since the heterogeneous network is not considered in [19]-[21], when simulation is performed under the heterogeneous network, SPTiRP is adapted such that it can be implemented to the heterogeneous network. The notations used in the remaining part of this section are listed as follows:

- $RN(\text{SPTiRP})$ and $RN(\text{SCA})$ represent the number of RNs deployed by SPTiRP and SCA, respectively.
- $RN(\text{SPTiRP}) - RN(\text{SCA})$ represents the difference of deployed RNs between SPTiRP and SCA.

A. Statistic Analysis

The number of RNs deployed by SPTiRP and SCA under different conditions is shown in Fig. 6, and the difference of deployed RNs between SPTiRP and SCA (i.e., $RN(\text{SPTiRP}) - RN(\text{SCA})$) under different conditions is shown in Fig. 7.

The results show that RNs deployed by SCA are fewer than those deployed by SPTiRP under different communication radii and delay constraints, which demonstrates the efficiency of SCA. It can be seen from Fig. 6(a) that when the communication radii of SNs and RNs is set as $r = 10$ and $R = 15$, SCA can save at most 5.4095 $(5.4095/24.9524 \approx 21.7\%)$ deployed RNs comparing to SPTiRP. In Fig. 6(b), the communication radii of SNs and RNs is set as $r = R = 20$, SCA can save at most 0.8108 $(0.8108/8.8108 \approx 10\%)$ deployed RNs comparing to SPTiRP. In Fig. 6(c), the communication radii of SNs and RNs is set as $r = R = 15$, and the maximum saving due to SCA is 2.18 $(2.18/17.5 \approx 12.5\%)$. In Fig. 6(d), the communication radii of SNs and RNs is set as $r = R = 10$, SCA can save at most 30.6667 $(30.267/96.1429 \approx 31.48\%)$ deployed RNs comparing to SPTiRP.

It is clear in Fig. 6 that the number of RNs deployed by SPTiRP and SCA both increases with the decreasing of communication radii of SNs and RNs, and it is easy to explain this by that more RNs are required to connected the whole network as their communication radii become smaller. Fig. 6 also indicate that as the delay constraint is relaxed, SPTiRP and SCA will deployed fewer RNs under the same communication radii. This is due to that as the delay constraint is relaxed, the algorithm will select the solution containing fewer RNs, even though this will yield feasible paths with larger hop count for the given SNs.

The number of RNs saved by SCA comparing to SPTiRP under different conditions is shown in Fig. 7. It can be learned from these figures that as the communication radii becomes smaller, more RNs can be saved by SCA comparing to SPTiRP, i.e., the value of $RN(\text{SPTiRP}) - RN(\text{SCA})$ grows.

This phenomenon can be explained by that as the communication radii becomes smaller, the topology becomes sparser and the hop count between the nodes on different paths becomes larger, therefore, the opportunity to merge different paths only by deleting the deployed RNs becomes smaller. Hence, the performance of SPTiRP deteriorates as the communication radii becomes smaller. On the other hand, SCA do not confront the limitation suffering by SPTiRP, therefore, the SCA algorithm will not suffer this deterioration. Consequently, the difference between these two algorithms grows larger as the communication radii becomes smaller.

Additionally, we can see from Fig. 7 that as the delay constraint is tighten SCA can save more RNs than SPTiRP. This is due to that SCA can deploy fewer RNs on each level based on the set covering approach even when delay constraint becomes smaller. In contrast, when delay constraint is tighten, the RNs can be deleted from the original shortest path tree by SPTiRP become fewer, which reduce the performance of SPTiRP. As a result, the difference of deployed RNs between SPTiRP and SCA becomes larger.

VIII. Conclusion

In this paper, the DCRNP problem is studied. For the NP-hard nature of the DCRNP problem, an approximation algorithm-SCA is proposed to handle this problem based on the set covering method. The novelty of the proposed SCA algorithm lies in its multi iteration procedure, and in each iteration the set covering method is applied to select the SNs or CDLs. The approximation ratio of the SCA algorithm is proved to be $\text{O}(\ln |S|)$. The time complexity of the SCA algorithm is also rigorously analyzed, and it shows that the SCA is a polynomial time algorithm with the time complexity of $\text{O}(N^3)$. Finally, extensive simulations are carried out to evaluate the performance of the SCA algorithm. The simulation results show that the SCA algorithm can significantly save the deployed relay nodes comparing to the existing algorithm within a worthy running time.

Acknowledgment

This work was supported by the Natural Science Foundation of China (61174026, 61233007, and 61304263) and Cross-disciplinary Collaborative Teams Program for Science, Technology and Innovation of Chinese Academy of Sciences (Network and System Technologies for Safety Monitoring and Information Interacting in Smart Grid).

References

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless Sensor Networks: A Survey,” *Computer Networks Journal*, vol. 38, no. 4, pp. 393-422, 2002.

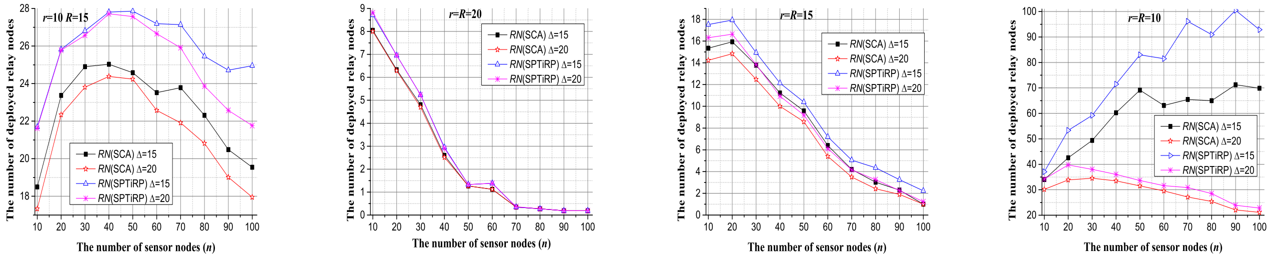
[2] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, “Next Century Challenges: Scalable Coordination in Sensor Networks,” *Proc. ACM MobiCom ’99*, pp.263-270, 1999.

[3] J. Yick, B. Mukherjee, and D. Ghosal, “Wireless sensor network survey,” *Computer Network*, vol. 52 n.12, pp.2292-2330, 2008.

[4] X.R. Bao, S. Zhang, D.Y. Xue, and N. Lin, “An Improved Centralized Voronoi Tessellation Algorithm for Wireless Sensor Network Coverage Problem,” *Information and control*, vol. 38, no. 5, pp. 620-623, 2009.

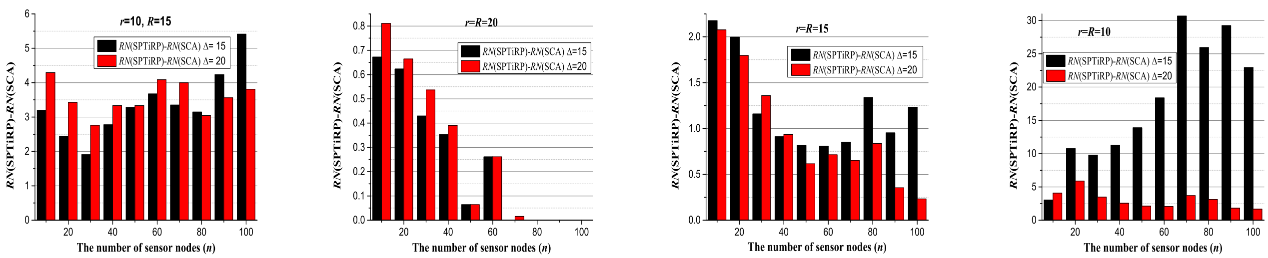
[5] H. Su, and X. Zhang, “Battery-Dynamics Driven TDMA MAC Protocols for Wireless Body-Area Monitoring Networks in Healthcare Applications,” *IEEE J. Sel. Areas Commun.*, vol. 27, no. 4, pp. 424-434, May 2009.

[6] SK. Chen, T. Kao, CT. Chan, CN. Huang, CY. Chiang, CY. Lai, TH. Tung, and PC. Wang, “A Reliable Transmission Protocol for ZigBee-Based Wireless Patient Monitoring,” *IEEE Trans. Information Technology in Biomedicine*, vol. 16, no. 1, pp. 6-16, Jan. 2012.



(a) $r = 10, R = 15$. (b) $r = R = 20$. (c) $r = R = 15$. (d) $r = R = 10$.

Fig. 6. The number of RNs deployed by SPTiRP and SCA under different r, R and Δs.



(a) $r = 10, R = 15$. (b) $r = R = 20$. (c) $r = R = 15$. (d) $r = R = 10$.

Fig. 7. The different of deployed RNs between SPTiRP and SCA under different r, R and Δs.

[7] V. C. Gungor, and G. P. Hancke, "Industrial Wireless Sensor Networks: Challenges, Design Principles, and Technical Approaches," *IEEE Trans. Ind. Electron.*, vol. 56, no. 10, Oct. 2009.

[8] A. A. Kumer S., K. Øvsthus, and L. M. Kristensen, "An Industrial Perspective on Wireless Sensor Networks-A Survey of Requirements, Protocols, and Challeges," *IEEE Commun. Surv. Tuts.*, vol. 16, no. 3, 2014.

[9] G. Lin, and G. Xue, "Steiner Tree Problem with Minimum Number of Steiner Points and Bounded Edge-length," *Inf. Proc. Lett.*, 69(2), pp. 53-57, 1999.

[10] D. Chen, D.Z. Du, X.D. Hu, G. Lin, L. Wang, and G. Xue, "Approximations for Steiner Trees with Mnimum Number of Steiner Points," *J. Global Optimization*, vol. 18, no. 3, pp. 17-33, 2000.

[11] X.Z. Cheng, D.Z. Du, L.S. Wang, and B.G. Xu, "Relay Sensor Placement in Wireless Sensor Networks," *Wireless Networks*, 14(3), pp. 347-355, 2007.

[12] J. Tang, B. Hao, and A. Sen, "Relay Node Placement in Large Scale Wireless Sensor Networks," *Computer Communications*, vol. 29, no. 4, pp. 490-501, 2006.

[13] E. Lloyd and G. Xue, "Relay Node Placement in Wireless Sensor Networks," *IEEE Trans. Comput.*, 56, no. 1, pp. 134-138, Jan. 2007.

[14] A. Srinivas, G. Zussman, and E. Modiano, "Construction and Maintenance of Wireless Mobile Backbone Networks," *IEEE/ACM Trans. on Netw.*, vol. 17, no.1, pp. 239-252, 2009.

[15] Q. Wang, K. Xe, G. Takahara, and H. Hassanein, "Device Placement for Heterogeneous Wireless Sensor Networks: Minimum Cost with Lifetime Constraints," *IEEE Trans. Wireless Commun.*, vol. 6, no. 7, pp. 2444-2453, 2007.

[16] S. Misra, S. Hong, G. Xue, and J. Tang, "Constrained Relay Node Placement in Wireless Sensor Networks to Meet Connectivity and Survivability Requirement," *Proc. of IEEE Infocom'08*, pp. 879-887, 2008.

[17] S. Misra, S. Hong, G. Xue, and J. Tang, "Constrained Relay Node Placement in Wireless Sesnor Networks: Formulation and Approximations," *IEEE/ACM Trans. Netw.*, vol. 18, no. 2, pp. 434-447, 2010.

[18] D.J. Yang, S. Misra, X. Fang, G.L. Xue, and J.S. Zhang, "Two-Tiered Constrained Relay Node Placement in Wireless Sensor Networks: Computational Complexity and Efficient Approximations," *IEEE Trans. Mobile Comput.*, vol. 11, no. 8, pp. 1399-1411, 2012.

[19] A. Bhattacharya, and A. Kumar, "Delay Constrained Optimal Relay Placement for Planned Wireless Sensor Networks," *IEEE Proc. IWQoS'10*, pp. 1-9, 2010.

[20] A. Bhattacharya and A. Kumar, "QoS Aware and Survivable Network Design for Planned Wireless Sensor Networks," *Tech. Rep.*, arxiv.org/pdf/1110.4746, 2013.

[21] A. Bhattacharya and A. Kumar, "A Shortest Path Tree Based Algorithm for Relay Placement in A Wireless Sensor Network and Its Performance Analysis," *Computer Networks*, vol. 71, no. 4, pp. 48-62, 2014.

[22] L. Sitanayah, K.N. Brown, C.J. Sreenan, "Fault-tolerant Relay Deployment based on Length-Constrained Connectivity and rerouting centrality in Wireless Sensor Networks," *9th European Conference on Wireless Sesnor Networks (EWSN)*, pp. 115-130, 2012.

[23] A. Nigam, and Y.K. Agarwal, "Optimal Relay Node placement in Delay Constrained Wireless Sensor Network Design," *European Journal of Operational Research*, vol. 233, no. 1, pp-220-233, 2014.

[24] S. Voss, "The Steiner Tree Problem with Hop Constraints," Annals of Operations Research, vol. 86, pp321-345, 1999.

[25] A.M. Costa, J.-F. Cordeau, and G. Laporte, "Fast Heuristics for the Steiner Tree Problem with Revenues, Budget and Hop Constraints," *European Journal of Operational Research*, vol. 190, no. 1, pp. 68-78, 2008.

[26] I. Gouveia, A. Paias, and D. Sharma, "Modeling and Solving the Rooted Distance-Constrained Minimum Spanning Tree Problem," *Computers & Operational Research*, vol. 35, no. 2, pp. 600-613, 2008.

[27] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, "Introduction to Algorithm," *The MIT Press and McGraw-Hill*, 2001.

[28] J.A. Bondy, and U.S.R. Murty, "Graph Theory," *Springer London Ltd.*, 2008.